心理学研究と電子計算機

美　濃　哲　郎

本論文では，電子計算機の歴史を概観し，まず，プログラム言語の問題点に触れる。次に心理学研究における電子計算機導入の意義，実験制御における諸問題について考察し，最後に今後の問題について考える。

I　Pascal から Neuman 型計算機まで

自然科学の歴史の中で，あるひとつの発明や発見がその分野だけでなく他の多くの研究分野にまで影響をあたえた例は多く見られる。古くは「零」の発見であり，また新しくは物理学における相対性理論がそうであった。それらの中には，我々の日常生活にまで変革をおよぼしたものもある。電子計算機もそのような発明のひとつであろう。しかし，Weizenbaum (1976) が指摘しているように，電子計算機は過去の発明とは異なった点をもっている。過去の発明は何らかの形でヒトの手や足の機能を延長したものであった。しかし，電子計算機はヒトの知的機能を補なったもので，明らかに過去のものとは性格を異にしている。この新しい発明から生まれてきた研究は今日も発展を続け，その成果は我々の日常生活に浸透しつつある。現在，自然科学は言うに及ばず行動科学においても，電子計算機の出現によって全く影響を受けていない分野はほとんど無いであろう。

計算機が最初に作られたのは，1642年，Pascal による歯車式加算機であるとされている。これは，今日の卓上電子計算機と基本的に同じものであり，いわゆる電子計算機の構造とは根本的に異なっている。自動計算機を最初に考案したのは1862年の Babbage であるとされているが，計算機の理論的基礎を与えた論文は

Turing (1937) によって発表されたものが最初であろう。しかし，実際にはその理論にもとづいた計算機は作られていない。Turing の業績は今でも「チューリングマシン」と呼ばれる仮想計算機[1] として，電子計算機の様々な問題を考える試金石になっている。1944年，イギリスの Aiken らによる MARK-I は，リレー式計算機と呼ばれ自動式計算機の最初である[2]。翌年の1945年に，今日の電子計算機の驚異的な発展をもたらすきっかけとなる「プログラム内蔵方式」を Von Neuman が提唱した。プログラム内蔵方式とは，計算の対象であるデータと計算手順を指示するコードを記憶装置内に平等に配置するという考え方である。この考え方によって，電子計算機は従来の単なる計算機から大きく性格を変えてゆくことができたわけである。一般的には，世界最初の電子計算機は1946年に Eckert と Mauchly による真空管式計算機 ENIAC であろう。ENIAC は真空管18000本，動作電力 140kW という大規模な装置であった。リレーから真空管による電子式にかわったことにより，10進10桁の加算が200μ 秒という演算速度でおこなわれるようになった。ただ，ENIAC は先に述べたプログラム内蔵方式ではなく，外部のジャンパ線によるものであった。プログラム内蔵方式の電子計算機は，1949年に Wilkes によって設計された EDSAC が最初であり，後に UNIVAC 1 として商品化されている。その後，電子計算機は演算素子としてトランジスタや集積回路等の発明によりいろいろな面で発展し，記憶装置の発展とともに小型化が進み，今日では ENIAC とほぼ等しい能力の電子計算機は手のひらに載ってしまうほどのものになった。

このような電子計算機のハードウェア[3] の発達とともに重要なのは，電子計算機を動作させる用法—ソフトウェア—の発達である。電子計算機は他の機械とは異って，その使用手段のノウハウに対する比重が大きい。電子計算機は片手で持ち運びできるが，そのソフトウェアを媒体[4] を用いて運ぼうとするとフォークリフトが必要だという話や，ハードウェアの価格よりもソフトウェアや使用者のための教育費の方がはるかに高いという事実がそれを証明している。

ソフトウェアの大部分は電子計算機のプログラムあるいはそのプログラムの作

成と関係がある。したがって，ソフトウェアとはすなわちプログラムとほぼ同義語と考えてよいだろう。そのプログラムとは電子計算機にある目的にかなった動作をさせるために，計算機にそなわった命令を組合わせた一連の命令の集まりである。このような命令は機械語と呼ばれ，それぞれの計算機特有の命令群をもっている。計算機内部では，データはすべて—命令コードも—2進数で表現されているので，機械語でプログラムを作る場合は，2進数による表現を用いなければならない[5]。そこでこのようなプログラムを少しでも扱いやすくするために，アセンブラと呼ばれる一種の記号を用いた言語が用いられるようになった。アセンブラは機械語の命令1つ1つに対応して記号がつけられ，その記号は機械語命令の動作を反映するような名前になっている。例えば，加算命令の機械語に対応したアセンブラ命令—記号—は ADD があてられる[6]。さらに，このアセンブラ命令で表現されたプログラムを機械語に変換する仕事を電子計算機にさせる事により，かなり大きなプログラムを作成することが可能になった。しかし，アセンブラがいかに理解し易くなったといえ，単なる記号の羅列に変わりはない。そこで命令としてある程度まで日常語に近い単語を用い，プログラムを書きやすくする試みがなされた。そうした努力によって作り上げられたのが高級言語とよばれるプログラム言語である。高級言語の1命令，例えば READ とか WRITE といった，に対応する機械語プログラムは数百ステップ以上であり，それゆえプログラムの作成が非常に楽になっている。そういった高級言語として最初のものが1956年，IBM 社によって開発された FORTRAN である。FORTRAN は今日の言語から見ると決して使いよい言語とは言えないが，初めての高級言語として世界中に拡まり，一種の電子計算機の共通言語となっている。高級言語がこのように拡まった理由のひとつは，アセンブラのように機械を意識せずにプログラムを作ることが出来ることである。従って，どんな電子計算機でも FORTRAN が使えたならば，FORTAN 言語で書かれたプログラムからはほぼ同じ結果が得られる[7]。FORTRAN 以後，高級言語としてヨーロッパで ALGOL が，米国国防省で COBOL が，というふうに次々と開発され，現在では非常に多くの言語が開

発されている。FORTRAN や ALGOL が科学技術計算用に，RPG や COBOL が事務処理用に，LISP や SNOBOL が言語処理用にと，高級言語は通常ある目的のプログラムを作成しやすいように開発されているが，PL/I のように汎用性を目指した言語もある。

II プログラム言語

前節で述べたようにハードウェア，ソフトウェア両面で発達してきた電子計算機は，心理学の分野にも少しづつ影響をあたえている。米国で「ミニコンピュータ」と呼ばれる小型電子計算機[8]の価格が1万ドルになって以来，ミニコンピューターが数多くの心理学研究室に導入されるようになった。そこで本論文では，このように電子計算機が身近になった心理学研究の現状において，計算機導入の意義，使用上の問題点および将来への展望について考察を加えたい。

電子計算機が考案された当初の目的のひとつに，昔，心理学研究でもよく用いられていた正確な数表の作成があった。しかし，実際に電子計算機が稼動すると，その数表は必要でなくなっていることに気がつく。なぜなら，求める数値を計算機に計算させればよいからである。このような例に似た現象は，電子計算機導入過程でしばしばみられる。多くの場合，これは使用者が考えている以上の能力を電子計算機が潜在的にもっているからである。したがって，我々は電子計算機に何ができるかということについて正確な知識をもつ必要がある。電子計算機はその名が示すように「演算装置」がその中枢となっているために「計算をさせるための装置」として利用されることが多い。事実，現在でも心理学研究の中で電子計算機がもっともよく利用されるのは，種々のデータ処理および解析の分野である（森本，1979)。おそらく多くの人々は電子計算機を「電卓」と呼ばれる卓上計算機の延長線上のものとして理解しているのであろう。しかし，今日，電子計算機が利用されている状況—身近な例として銀行のオンラインシステムや国鉄の「緑の窓口」がある—を考えてみた場合，その可能性ははかりしれないほど大き

い。

データ処理の目的のためには統計用のプログラムが数多く用意されている。このようなプログラムは先に述べた高級言語とは区別されて，「統計パッケージ[9]」と呼ばれている。統計パッケージは計算機の知識がほとんどなくても誰でも使用することができる。使用者はデータ，統計処理名，そしていくつかのパラメータを入力するだけで，詳細な統計処理の結果がプリントアウトされてくる。このような統計処理については批判もあろうが，電子計算機の使用目的のひとつが理想的に達成された一例であろう。通常このような統計パッケージは大学の計算センター等に用意されていて，研究室内のミニコンピュータで稼動するような統計パッケージは，一部のミニコンピュータ用を除いてほとんど用意されていないのが現状である。とくに，心理学のような行動科学領域で使用できるミニコンピュータ用の統計パッケージはほとんど作られていない。そこで，研究者が高級言語で統計処理用のプログラムを作成する場合がしばしば見られる。このような場合，電子計算機に特有な問題のひとつ[10]が重要になってくることがある。具体的な例として，我々が統計でよく用いる標準偏差の算出を電子計算機のプログラムとして表現する場合を考えてみよう。

標準偏差を計算する場合，次の2つの式を我々は知っている。Nはデータの個数，Xiは個別データ，$\bar{X}$は平均値である。

$$SD=\sqrt{\frac{\sum_{i=1}^{N}(Xi-\bar{X})^2}{N}} \quad \cdots\cdots(1)$$

$$SD=\sqrt{\frac{\sum_{i=1}^{N}Xi^2}{N}-(\bar{X})^2} \quad \cdots\cdots(2)$$

式(1)は理論式で，式(2)は式(1)を変形し，手計算に便利なようにしたものである。そして，我々は常に式(2)の方を使用するために，プログラムを作成する場合，式(2)に従った表現を用いてしまう。図1は，式(2)にもとづいて作られたFORTANプログラムである。心理学研究者用に書かれた書物の中で，標準偏差のみならず，

```
FORTRAN IV       V01C-03A

      C      PROGRAM 1
      C
0001          SUM=0.0
0002          SQRSUM=0.0
0003          READ(5,100) N
0004      100 FORMAT(I5)
0005          DO 10 I=1,N
0006          READ(5,102) X
0007          SUM=SUM+X
0008          SQRSUM=SQRSUM+X**2
0009      102 FORMAT(F10.3)
0010       10 CONTINUE
0011          DMEAN=SUM/FLOAT(N)
0012          SD=SQRT(SQRSUM/FLOAT(N)-DMEAN**2)
0013          WRITE(6,200) DMEAN,SD
0014      200 FORMAT(1H0,3X,8HMEAN = ,F10.3,8H    SD = ,F10.3)
0015          STOP
0016          END
```

図1　式(1)にもとづいた FORTRAN プログラム

分散分析や他の多くの統計処理のプログラムに対して採用されている計算式は式(2)のタイプのものが多い（田中良久，1975)。このようなプログラムを用いて多量のデータを処理した場合，手計算による結果と異った結果が得られる場合がある。また，極端な場合には計算機で実行中にエラー[11]が生じて計算できなかったり，分散分析のF値が負数になったりすることがある。これには電子計算機内部で数がどのように表現されているが関係している。通常，電子計算機で実数を扱うとき 10^{-37} から 10^{38} までの値を扱えるが，有効桁数は10進で6桁ないし7桁のため，市販されている卓上電子計算機より精度が悪い。すなわち，大きな数に小さな数を加減すると桁おちを生じ，その小さな数は実際上は電子計算機によって演算されていないことになる。しかも，エラーメッセージ[13]はあらわれない。ここで，前述の式にもどってみると，式(1)と式(2)を比較した場合，式(2)の方が大きい数と小さな数を加算する可能性が高い。なぜなら式(2)の右辺前項は2乗の総和を求めている。図1のプログラムで示すなら第8行の演算である。従って，このようなことが生じないようにするために，式(1)にもとづいた図2のようなプログラムを

```
FORTRAN IV      V01C-03A

      C     PROGRAM 2
      C
0001        DIMENSION X(100)
0002        SUM=0.0
0003        SQRSUM=0.0
0004        READ(5,100) N
0005    100 FORMAT(I5)
0006        DO 10 I=1,N
0007        READ(5,102) X(I)
0008        SUM=SUM+X(I)
0009    102 FORMAT(F10.3)
0010     10 CONTINUE
0011        DMEAN=SUM/FLOAT(N)
0012        DO 20 I=1,N
0013     20 SQRSUM=SQRSUM+(X(I)-DMEAN)**2
0014        SD=SQRT(SQRSUM/FLOAT(N))
0015        WRITE(6,200) DMEAN,SD
0016    200 FORMAT(1H0,3X,8HMEAN = ,F10.3,8H   SD = ,F10.3)
0017        STOP
0018        END
```

図2 式(2)にもとづいた FORTRAN プログラム

作成すべであろう。おそらく，先に述べた統計処理用のプログラムを掲載している書物の著者は，語長[13]の長い中型あるいは大型計算機で掲載するプログラムを実行させて誤りのない事を確めたのであろう。しかし，それを見て使用する側は語長の短い計算機を用いる場合が多いので，小型の計算機で実行する場合に桁落ちが生じる可能性があることを記載する配慮が必要であるだろう。また，プログラムを印刷物として発表する場合の問題について安田（1978）は，媒体として，コストは安いが，誤植という大きな問題があると述べている。事実，先に述べた田中（1975）の中にもプログラムをそのまま実行させたならば，掲載されているような結果を出力してこないものが含まれていた。このように電子計算機の使用においては，小さな誤りやソフトウェアの無理解によってもたらされる障害は大きい。

従来，心理学研究で用いられてきた実験装置に関しては，その装置の機能を達成するために必要な電気回路や機械的な仕組みについて，研究者は知識を特に持

つ必要はなかった。最近のエレクトロニクスの発達による集積回路を用いた装置はその「ブラックボックス化」が激しい。生体アンプを用いる時は微弱な生体信号の増幅，タイマーでは動作時間の測定がそれらの装置の働きであることを知っていれば，装置内部にどのような回路が用いられているかを知らなくとも実験実施の上でなんら支障はなかった。このような装置とは異った型の装置として近年あらわれたものに，「ランダム-ロジック-コントローラ」がある。これは，いくつかの論理回路をワイヤーで結線し，実験を制御しようとする装置であるが，この結線の仕方によって装置は様々な働きをする。したがって，この装置は従来のものより高い自由度をもっているが，その代わりに実験者はその内部での働きをよく理解していなければならない。論理回路の結線—電子計算機のプログラムにあてはまる—をおこなうのは実験者であるので，実験結果に不審な点があればその原因を容易に解明することができる。

電子計算機は，上述のランダム-ロジック-コントローラよりもはるかに高い自由度をもっている。なぜなら，後者はプログラムがプログラム自体を変更することはできないが，前者はそれが可能であり，そのことがいかに自由度を高めるかは想像に難くない。このように自由度の高い装置を使用する場合，その内部構造についての知識はいま述べたように重要であるが，実際に電子計算機の構造について詳細な知識を得ることは困難である。しかも，電子計算機の場合，ランダム-ロジック-コントローラと異り，そこで使用するプログラムも多くの場合，他人の作ったものを用いなければならない。FORTRAN や BASIC[14] といった高級言語を形成している機械語命令は非常に大きなステップ数であるので，おそらくその内容を細かに理解することは不可能であろう。しかし，使用者は単にその言語の文法を知っているだけで良いとは言えない。そういった言語がいかなる構造をもっているかといったことや，いかなる目的で開発された言語であるかを知ることは重要であろう。

通常，高級言語にはコンパイラとインタプリタの2種類がある。FORTRAN や COBOL はコンパイラであり，BASIC や APL[15] はインタプリタである。コ

ンパイラはその言語で書れたプログラムを一度に全て機械語に変換したり，ある種のコードに変換したりするが，インタプリタは実行時に実行すべき命令を解釈しながら命令に従った動作をする。こういった方式の差は，例えばプログラムの実行速度にあらわれてくる。

次に，いま述べた方式の差異にも関係している事であるが，こういった高級言語が開発された目的を知ることは，プログラム作成上の効率や実行時の速度を考えた場合，重要になってくる。

FORTRAN 言語は，現在，多くの分野で利用されており，電子計算機のプログラムを教育する場合ほとんどこの言語が採用されている。そのため，初心者のなかには，電子計算機と FORTRAN を同義語と解釈しているものもある。このような傾向は FORTRAN を学べば，どこの電子計算機ででもプログラムを作成することができるという利点はあるにしても，電子計算機のもつ可能性のほんの一部しか利用していないことになる。

先に述べたように，高級言語はある目的に便利なように開発されている。FORTRAN や BASIC は科学技術計算用に，RPG や COBOL は事務処理向きに，LISP[16] は言語処理用に開発されている。また，心理学と関係ある言語としていま述べた LISP の他に，SKED (Snapper と Kadden, 1973) や ACT (Millenson, 1973) 等があり，これらは特に実験制御に向いた言語である。これについては後の節で詳しく述べる。このように種々の目的に応じた言語があるにもかかわらず，ほとんどの電子計算機教育が FORTRAN にもとづいておこなわれているためか，どのような目的にも FORTRAN 言語が用いられている。また，最近は BASIC 言語がとくにマイクロコンピュタ[17] の流行とともに広まったために，本来は TSS 用言語であるのに，この言語を用いて実験を制御しようという試みがいくつか報告されている（苧阪，1979)。

先に述べたように，電子計算機は非常に自由度の高い「道具」であるために，本来の目的とは異った目的に高級言語を使用することが充分可能である。しかしながら，そういった使用法では，i）処理速度がおそくなり，ii）適当な命令が用

意されていないために，プログラムが大きくなり，iii）出来上がったプログラムの表現が煩雑でわかりにくくなる，といった問題が生じてくる。特にプログラムの表現については，Dijkstra（1972）がプログラミングの方法に関わる基本的な問題を述べるなかで，わかりやすいプログラム表現の重要性を挙げている[10]。

今日では，様々な目的に応じた種々のソフトウェアが開発されているので，それらの中から目的にあったものを選択し使用する事が望ましい。

Ⅲ　心理学と電子計算機

それでは，電子計算機というあたらしい「道具」は心理学の研究に何をもたらしてくれるであろうか。ここで言う「道具」とは，先に何度も述べてきたことから明らかなように，電子計算機というハードウェアではなく，もっと重要なソフトウェアのことを示す。心理学の歴史においても，新しい「道具」の開発によって，研究の進展が大きく影響を受けた例は多い。化学の研究における「周期率」の発見は Wundt の心理学に影響をおよぼし，量的変化の測定技術の向上は直接にあるいは間接的に行動主義の出現に影響したであろう。「スキナー箱や累積記録機」は Skinner（1938）の「行動分析」という考え方を支えたであろう。もちろん，それぞれの心理学は正当な理論的背景をもって出現してきたのであろうが，こういった自然科学史における現実的な面を見逃すわけにはいかない。物理学や化学における発展は，「道具」の発展に負う所が大きいが，心理学もその例外ではないだろう。

心理学に数学モデルがとり入れられたのは，かなり以前である。Gullicksen（1934）によれば，1907年にすでにその試みがなされている。その後，1930年に Thurston が確率の概念を学習のモデルにとり入れて以来今日まで発展してきている。こういったモデルから，そのシュミレーションを実際におこなってくれるのが電子計算機である。手計算の時代では不可能であったような繰返し計算も短時間で実行してくれ，速やかに結果を得ることが出来るようになった。また，モ

デルをプログラムに表現することによって新しい問題点を発見することもあるであろう。このような例として有名な Feigenbaum (1963) の無意味つづりの記憶モデルを挙げることができる。

彼のシュミレーションプログラムに学習をおこなわせてみると，言語学習にみられる種々の性質を示し，特にリストAを学習させたのちリストBを学習させると，リストAの再学習が妨害されるという，逆行抑制の現象が再現された。また，短期記憶に関して当時は知られていなかった機構の必要性が見出され，プログラムには組み込まれていないような「行動」が示された。このようなシュミレーションプログラムでは，種々の性質を一種のサブルーチン[10]として表現するわけであるが，そのようなサブルーチンで表現していなかった「干渉」現象を彼のプログラムが示したのである[10]。

Newell と Simon (1972) に代表されるいわゆる「情報心理学」は，さらに電子計算機の発展に影響を強く受けた分野であろう。これは一種の人間機械論的な考え方ではあるが，ヒトをひとつの情報処理機構とみなしその性質を明らかにしようとするものである。この立場をとることによって「認知」という問題を扱いやすくし，Neisser (1967) や Rumelhart (1977) のような「認知心理学」を発展させてきている。また，情報心理学の分野で生まれた言語として Bobrow と Fraser (1969) の ATN がある。最初は，自然言語の計算処理のプログラムとして開発されたが，後にヒトの言語理解のモデルとして応用されている。

最近，研究が盛んになりつつある分野に「人工知能」がある。この分野では様々なプログラムが開発されており，チェスをするプログラムが米国のある州のチャンピオンになったという話もある。しかし，有名なものとしては Newell と Simon の GPS がある。これは，一般問題を解決する (General Problem Solver) という言葉の頭文字をとったものあるが，その名前ほどの成果はあがっていない。人工知能に関するプログラムで最も注目を集めたもののひとつに Weizenbaum が開発した「ELIZA」がある。彼は MIT の電子計算機科学科の教授であるが，精神科医の役目をする「ELIZA」と名づけられたプログラムを開発した。図3に

Men are all alike.
In What Way ?
They 're always bugging us about something or other.
Can you think of a speccific example ?
Well, my boyfriend made me come here.
Your boyfriend made you come here.
He says I'm depressed much of the time.
I am sorry to hear you are deprcssed.
It's true. I am unhappy.
Do you think coming here will help you not to be unhappy ?
I need some help, that much seems certain.
What would it mean to you if you got some help ?
Peshaps I could learn to get along with my mother.
Tell me more about your family.
My mother takes care of me.
Who else in your family takes care of you ?
My father.
Your father.
You are like my father in some ways.
What resemblances do you see ?
You are not very aggressive, but I think you don't want me to notice that.
What makes you think I am not very aggressive ?
You don't argue with me.
Why do you think I don't argue with you ?
You are afraid of me.
Does it please you to believe I am afraid of you ?
My father is afraid of everybody.
What else comes to mind when you think of your faher ?
Bullies.
Does that have anything to do with the fact your boyfriend made you come here ?

図3　ELIZA と患者の会話 Weizenbaum (1978) より

「ELIZA」と患者の「会話」例を示す。「会話」は，ブラウン管ディスプレイとタイプライターの間でおこなわれる。斜体字の部分が患者からの入力であるが，あたかも電子計算機が患者のコトバを理解しているかのように見える。Weizenbaum は，「ELIZA」に対面する多くの人々に電子計算機に対する過信を警告しているが，重要なことは電子計算機に対する正しい理解であろう。現在，ハードウェアの面における発展は目覚しいものであるが，ソフトウェアの発展は非常に遅

れている。

情報処理心理学や人工知能の研究はまだ始まったばかりである。それらの研究者は電子計算機について常に新しい知識を吸収してゆかねばならない。現在，電子計算機の「割り込み」や，「DMA」あるいは「記憶領域管理(24)」やその他枚挙しきれない機能を少しでも取り入れたモデルはあまりないようである。これらを使わずに，情報処理モデルや人工知能を過信したり批判するのは時期尚早ではないであろうか。心理学者は，工学者とは異った観点からこれらの機能を取り入れた研究が出来るようにおもわれる。

Ⅳ　実験制御における諸問題

最後に，電子計算機を実験制御に用いる場合の問題について触れる。タイマーやリレー，トランジスタや集積回路あるいはロジック-コントローラを用いておこなってきた従来の実験制御を電子計算機でおこなえば，種々の理由で非常に便利である。Cleary (1977) も述べているように，実験の実施が容易で，得られたデータの処理も速やかにおこなえ，人の手が介在しないので間違いも少ない。データを人手で電子計算機に打ち込んでデータ処理をおこなう場合，データ量が少ない場合は問題はないが，1000以上になるとエラーチェックのすべもなく，必ず入力ミスはつきまとう。このように考えた場合，電子計算機で統計処理をおこなう利点を生かすには，如何に人手の介在をなくすかが重要となる。その点，秋田 (1978) や苧阪 (1979) も述べているように，実験制御を電子計算機でおこなうことはデータが集収から処理まで電子計算機内で扱かわれるので最適の方法であろう。

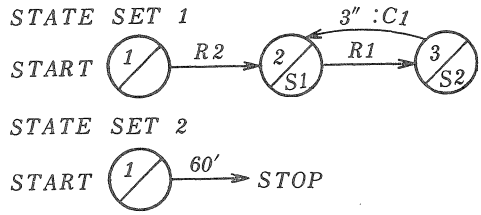
初期の実験制御は機械語やアセンブラのプログラムによっておこなわれていた。今日でも，マイクロコンピュータで実験制御をおこなう時にこの方法が用いられることもある。機械語やアセンブラによるプログラムは作成に時間がかかり，また誤まりなしにそのプログラムを働かせるまで非常に労力がいる。この作業は精

神的苦痛が大きく，しばしば実験者を悩ませる。その上，重要なことであるが，書かれたプログラムから如何なる実験を制御しているのか理解することは全く不可能であり，あるていど期間が過ぎると実験者自身もそのプログラムが作られた当初の目的を忘れてしまうほどである。

次にあらわれたのは，高級言語である FORTRAN や BASIC を少し変更して実験制御プログラムを作成する方法である。この方法ならば，プログラム作成のための労力は軽減される。しかし，この方法でも問題はいくつか残る。まず第1に，FORTRAN や BASIC は本来科学技術計算のために開発された言語であるために，方程式の表現や処理，あるいは計算精度といった面に注意が払われ，入出力の時間的な関係，時間測定や時間経過に伴なう処理のための命令は用意されていない。実際，入出力よりも計算処理の方に優先度をもたせている場合が多いようである。従って，このような仕様の言語を用いて実験制御をおこうこと自体に問題があるが，さらに FORTRAN や BASIC で作成されたプログラムが実験をうまく表現していないという問題がある。先に挙げた秋田や苧阪は BASIC 言語で表現された実験制御プログラムを発表しているが，実際，この言語に通じているものでも何をおこなっているプログラムかを理解することは困難である。それは，プログラム中に本来 BASIC にはない命令である WRITE や POKE といった命令が用いられているからである。おそらく，外部機器のスイッチを制御するためのものであろうが，その命令語自体によって何ら実験制御と関連した事象が表現されていない。このようなプログラムの中では，Cleary (1977) の BASIC 修正言語は比較的わかりやすい表現が用いられているが，実際にそのような言語は現在のところ作らていない。

電子計算機のメーカーやソフトウェア産業は[8]，実験制御用の高級言語を適切に供給していない。おそらく，そのような言語を開発しても労力に見合うだけの採算がとれないからであろう。しかし，心理学者自身の手で制御用高級言語がいくつか開発されてきている。最も有名なものとして Snapper と Kadden (1973) による SKED がある。彼らはとくに，オペラント条件づけの実験制御に有効な言

語として SKED を開発した。同時に，Snapper, Knapp と Kushner (1970) は強化スケジュールを厳密に表現するための状態図の概念をとり入れた。この概念，は，先に述べた ATN のものとよく似た概念であるが，SKED によって実験をする場合，実験者は実験手続きを状態図に表現し，それを SKED 言語に置き換えてゆく。SKED 言語は状態図をそのまま直接表現できるように作られている。図4に連続強化スケジュールの状態図と SKED のプログラムが示されている。



State diagram for continuous reinforcement schedule. The number of reinforcements is counted by C1 and the parallel state set 2 is used to terminate the session after 1h. S1 is the keylight, and S2 provides reinforcement. R1 is the response key, and R2 is used to start thesession. The units of time are denoted by '[minutes] and "[seconds].

```
/CONTINUOUS REINFORCEMENT SCHEDULE
S. S. 1,
S1,
        R2 : ON1 ·····→S2      /START SESSION
S2,
        R1 : OFF1 ; ON2 ·····→S3      /GIVE REINFORCEMENT
S3,
        3" : OFF2 ; ON1 ; C1 ·····→S2
S. S. 2,
S1,
        6O' ·····→STOP /SESSION TIMER
        $
```

図4　連続強化スケジュールの状態図と SKED プログラム［Clearly (1977) より］

SKED 言語は強化ンケジュールの研究において便利で有効であるが，あまり一般的なものではない。Millenson (1973) の ACT 言語は SKED より表現が明確に

なっている。ACT 言語の特徴は，表現が英文に近く記述しやすいことと，round-robin 方式と呼ばれる一種の TSS で，複数のオペラント条件づけ実験を同時に制御できることにある。また，実験制御のあき時間には他のプログラムを実行できるように設計されている[23]。

我々が開発した TYMES 言語（瀧川，1978；Takigawa と Mino, 1980）は SKED や ACT より自由な表現ができるようになっている。この言語の場合はとくに状態図を考える必要はなく，命令も実験制御に関連した語が使用されており，また変数名として自由に名前をつけることができるので実験内容に即した表現が可能である。TYMES も ACT と同様に複数個の実験を制御することができるし，また制御の時間以外の空き時間には FORTRAN や BASIC を用いてデータ処理をおこなうことができる。

実験制御を電子計算機を用いておこなう場合，秋田が述べているように，複雑な諸条件の設定をキーボードから打ち込むだけで1人で実験ができるという点は，科学者にとって魅力あるものである。しかし，BASIC や FORTRAN で実験制御をおこなう場合には先に述べた理由で，あまり生産的でない。これは，Rozenbaum が電子計算機を用いて統計処理をおこなっても何ら生産的でないと述べた言葉がそのままあてはまるように思う。

実験制御用語を用いて実験制御をおこなう潜在的効用は，かつて統計学が，あるいは実験計画法が心理学にもたらした効用と同様であるように思われる。統計学は，実験手続きでとられた処置がどのような効果をもたらしたかを客観的に判断させる根拠を与えてくれる。もし，統計処理法がなく，ただ単に結果を記述するだけにとどまっておれば，心理学の今日の姿はかなり異ったものになっているであろう。実験の結果については客観的判断の根拠があるにもかかわらず，手続きについては今なお記述に拠っている。このような記述式の手続きでは，論文から実験内容が詳しく読み取れない場合がしばしば見られる。しかし，実験手続きを実験制御言語で記述されたプログラムとして表現したならば，非常に客観的な表現になるのでないだろうか。そのプログラムで用いられる諸命令は，それぞれ

の言語仕様によってあらかじめ決定されている厳密な時間的，空間的動作と密接に結びついている。そのため，実験の再現は非常に容易となる。また，実験をこのようにプログラムとして表現した場合には，従来なら見過してしまうようなかくれた要因をはっきりと浮びあがらせてくれる。たとえば，実験手続きとして刺激提示順序や試行間間隔を決定するとき，論文に記述するときは「無作為順序」とか「刺激提示間隔は無作為で平均30秒であった」という表現が用いられるが，具体的には「10試行以内にすべての種類の刺激が提示され」たり「20秒，25秒，30秒，35秒と40秒の5種類の試行間間隔が無作為に選択され提示され」ていたりする。また，一度決定された刺激提示系列は繰返し，どの被験体にも使用されることが多い。実験制御言語で刺激提示順序をプログラムとして表現するときは，具体的に「10試行以内にすべての種類の刺激が提示される」ように書かなければならない。また，一度決定された刺激提示系列を繰返し用いるときは，プログラムに一種の Table として表現され，繰返し使用されているのが明白になる。

実験制御言語を用いて実験を行ったときに可能となる手続きとして，実験中に得られた反応をただちに処理したり，数値を変換したり，さらにそのようにして得られたデータを基準として次の刺激事象を決定したりすることができる。この事実は，実験事態とは分離された形で考えられてきたシュミレーションやモデル式を実験事態に導入することが可能であることを示す。こういった形で実験が実行されたとき，心理学実験は新しい姿を示しはじめるかもしれない。

V おわりに

現在，我々が手に入れることができる電子計算機はすべてノイマン型の計算機である。情報心理学や人工知能あるいは実験制御の研究者達は，この型の計算機の知識にもとづいて研究をおこなっている。しかし，ノイマン型電子計算機のハードウェアとソフトウェアの限界が問題になっている。ハードウェアの面では，真空管時代には利点であったノイマン型の single control flow stream という

構造が，大規模集積回路で電子計算機が構成されるようになると，逆に欠点となる。並列処理が可能なシステムが要求される現在では，ノイマン型計算機はその方向への発展の障害となってきている。また，ソフトウェアの面でも，single control flow stream 構造で必須である「GOTO」型命令によるプログラム構造上の弊害が重要な問題になってきている。ノイマン型電子計算機でのこのような諸問題を改良するために，データフロー型計算機や再帰型計算機などいくつかの種類の非ノイマン型電子計算機[9]が開発されつつある。

おそらく近い将来，いくつかの非ノイマン型の電子計算機が市場に現れるであろう。そのような計算機が手に入った時，プログラム作成上の思想がノイマン型とはまったく異なるので，上に述べた研究は大きくその内容が変わってくるにちがいない。したがって，他の分野の研究者と同じ様に，心理学の研究者も計算機科学の成果について無関心ではいられない。

かって，Neuman (1975) が計算機理論のひとつとして「自己増殖理論」を発表し，その思想が生命科学の分野に「遺伝子情報」という考えをもたらして後の研究発展に役立った。将来，すぐれた非ノイマン型計算機理論が発表され，それが心理学研究のまったく新しい方向への進歩に寄与するかもしれないと考えるのは，私だけの単なる幻想であろうか。

本論文作成にあたり，ご指導下さった関西学院大学文学部教授，宮田洋先生に深く感謝します。

註

(1) 仮想計算機とは，実際には存在しないが，計算機の機能を定義することによって，あたかも実在するかのように考えられた理論上の計算機である。新しい電子計算機の設計をおこなうとき，それを仮想計算機としてプログラムを作りその能力を推測する。

(2) MARK-I 以前にも，1887年に Hollerith によるパンチカード式統計機の発明があるが，いわゆる電子計算機のカテゴリには入っていない。

(3) ハードウェア (Hardware) とは，電子計算機の機構あるいは機械を示す語であり「金物」という訳が付されることもある。ハードウェアに対して，ソフトウェア (Software) とは電子計算機の使用技術を示す語である。

(4) ソフトウェアの媒体として，パンチカードや紙テープが知らているが，現在では，磁気テープや磁気ディスクなど記憶密度の高いものが用いられている。

(5) 計算機の命令は2進数16桁ないし32桁で成り立っているので読みづらい。実際上は8進数や16進数による表現が用いられる。

(6) 現在では，マクロ命令と呼ばれる，2つ以上のアセンブラ命令をまとめて，仮想の命令を作れるように工夫されていて，プログラムを作成しやすいようになっている。

(7) 個々の電子計算機内での数の表現の仕方によって精度が異ってくるので，結果が少し異ってくることがある。

(8) ミニコンピュータとは，16 bits の命令語長をもち，主記憶装置として 32K 語のコアメモリの他に外部記憶装置をそなえているものをいう。

(9) 心理学研究と関係がある統計パッケージとしては，BMD (Dixon, 1970), SPSS (Nie, Hull, Jenkins, Steinbrenner, & Bent, 1975) や SAS (Barr, Goodnight, Sull, & Helwig, 1976) がある。

(10) Forsythe ら (1974) は計算機科学の入門者が学ぶべき重要な点のひとつとして，実数表現のひとつである浮動小数点システムでの数値の近似とまるめ誤差や打切り誤差の発生および伝播を挙げている。

(11) 高級言語では，使用者がおかす文法上のまちがいやプログラム実行中にゼロによる除算をするといった計算を続けるには支障をきたす事態を知らせる機能をもっており，エラーチェック機能とよぶ。

(12) エラーが生じたとき，それを使用者に知らせる文をエラーメッセージと呼ぶ。

(13) 個々の電子計算機が基本的に取り扱う2進数での桁数 (bit) を語長とよぶ。

(14) Dartmoth 大学で開発された高級言語で，小型の FORTRAN 言語である。本来，TSS (Time Searing System：時分割システム) を目的としていて，会話型言語でもあるため，初心者が電子計算機でプログラムを学習するのにむいている。現在，マイクロコンピュータ用の高級言語の主流にもなっている。

(15) やはり会話型高級言語で，数学的な問題を解決するのにむいている。

(16) LISP はインタプリタとコンパイラの両方が用意されている。とくに言語処理能力がすぐれているので人工知能の研究によく用いられている。

(17) 計算機の高度な集積化によって生まれた超小型電子計算機である。たいてい語長は 8 bits であり，少しの周辺機器だけで稼動し，価格も低いため一般家庭にまで拡まりつつある。また，家庭電器製品に組込まれて，種々の自動制御に役立っている。

(18) プログラム作成上の誤りをふせぐためには，構造がはっきりとわかるプログラムを書くべきであると主張し，FORTRAN よりも ALGOL のような手続き言語が構造的プログラミングにむいていると述べている。

(19) プログラムの中で何度も同じような手続きが表現される場合，そのような表現をサブプログラムとしてまとめ，主プログラムから呼び出せるような形にできる。このサブプログラムをサブルーチンとよぶ。

(20) 通常，プログラムを実行させたならどのような結果がもたらされるか使用者は知っていると思われがちであるが，必ずしもそうではない。

(21) 「割り込み」は，あるプログラムの実行中に外部機器からの要求によって，プログラム実行を中断し，その機器の入出力に関するプログラムを実行することをいう。それが終了するともとのプログラムを続行することができる。外部機器によらない，ソフトウェアによる割り込みもある。DMA (Direct Memory Access) は，プログラムが主記憶装置を使用していない間に，直接外部機器と記憶内容をやりとりする機能で，ときには，プログラムの実行を一時的に遅らせて，「DMA」を実現することもある。
「記憶領域管理」とは，あるプログラムが使用できる記憶領域を制限できる機能をもった電子計算機で，いくつかのプログラムの使用領域をソフトウェアで管理させること。

(22) 電子計算機を生産したり販売せずに，電子計算機で使用するプログラムを開発しそれを販売する産業。

(23) ACT のあき時間に実行させるプログラムは，ACT と同時に稼動しているように見える。これは，ACT が実験制御を必要とする時，他方のプログラムに割込みがかかってプログラムの実行が ACT に移行するという形で実現されている。

(24) 非ノイマン型コンピュータについては，Backus (1978) に詳しい説明がなされている。

References

秋田宗平，江島義道「オンライン方式による視覚情報処理装置の試作研究」第42回日本心理学会発，1978.

Backus, J, Can programing be liberated from the von Neuman style ? A functional style and its algebra of programs. CAM 1978, 21, 8, 613-641.

Barr, A. J., Goodnight, J. H., Sull, J. P., & Helwig, J. T. A user : s guide to SAS76. Raleigh, N. C. : Spark Press, 1976.

Bobrow, D., & Fraser, B. An argumental state transition network analysis procedure. In Walker, D., & Norton, L. (Eds.) Proceedings of the International Joint Conference on Artificial Intelligence. Washington, D. C. 1969.

Clearly, A. Instrumention for Psycholog. New York : John Wiley & Sons, 1977.

Dijkstra, E. W. (野下活平等訳)「構造化プログラミング」サイエンス社，1972.

Dixon, W. J. BMD : Biomedical computer programs, Berkley : University oy Cal-

ifornia Press, 1970.

Feigenbaum, E. A. The simulation of verbal learning behavior. In Feigebaum, E. A., Feldman, J. (Eds.) Computers and Thought. New York : McGraw-Hill, 1963.

Forsyth, A. I., Keenan. T. A., Organick, E. I., & Stenberg, W. Computer Science. New York : Wiley, 1975.

Gullicksen, H. A rational equation of th learning curve based on Thorndike's law of effect. Journal of General Psychology, 1934, 11, 395-424.

Millenson, J. R. On-line sequntial control of experiments by an automated contingency transrator. In Weiss, E. (Ed.) Digital computers in the behavioral laboratory, New York : Appleton-Century-Crfts, 1973.

森本正昭「情報心理学」誠信書房，1979.

Neisser, U.（古崎敬，村瀬晏共訳）「認知の構図」サイエンス社，1978.

Newell, A., & Simon, H. A. Human problem solving. Englwood Cliffs, N. J. : Prentice Hall, 1072.

Nie, N. H., Hall, C. H., Jenkins, J. G., Steinbrenner, K., & Bent, D. H. Statistical Package for the Social Sciences (SPSS). New York : McGraw Hill, 1975.

苧阪直行「マイクロエンピュータを利用したリアルタイム BASIC による知覚実験の制御（RBLA）について，第91回関西心理学会発表，1979.

Rumelhart, D. E.（御領謙訳）「人間の情報処理」サイエンス社，1978.

Skinner, B. F. The behavior of organisms : An experimental analysis. New York : Appleton, 1938.

Snapper, A. A., & Kadden, R. M. Time-sharing in a small computer based on behavoral notation system. In Weiss, B. (Ed.) Digital computers in the behavioral laboratry. New York : Appleton-Century-Crofts, 1973.

Snapper, A. G., Knapp, J. Z., & Kushner, H. K. Mathematical description of schedules of reinforcement. In Shoenfeld, W. N. (Ed.) The theory of reinforcement schedules. New York : Appleton-Century-Crofts, 1970.

瀧川哲夫，実験制御モニター TYMES/PTS 親和女子大学研究論叢第11号，1978.

Takigawa. T., & Mino, T. TYMES : A high level language for process control and deta manipulation in the behavioral laoratory.

田中良久「BASIC」入門東京大学出版，1975.

Turing, A. M. Computability and λ-Definability. The Journal of Symbolic Logic, 1937, 2, 153-163.

Von Neuman（高橋秀俊監訳）「自己増殖オートマトン理論」岩波書店，1975.
Weizenbaum, J.（秋葉忠利訳）「コンピュータ　パワー」サイマル出版会，1976.
Weizenbaum, J. Moral Considerations of Artificial Inteligence. Personal Computing, 1978, 2, 21-29.
安田寿明「マイクロコンピュターその20」bit, 1978, 8.

——大学院博士課程後期課程——